Real-Time Edge Tracking Using a Tactile Sensor

Alan D. Berger, Richard Volpe and Pradeep K. Khosla
Department of Electrical and Computer Engineering
The Robotics Institute
Carnegie-Mellon University
Pittsburgh, PA 15213

Abstract

Object recognition through the use of input from multiple sensors is an important aspect of an autonomous manipulation system. In tactile object recognition, it is necessary to determine the location and orientation of object edges and surfaces. We propose a controller that utilizes a tactile sensor in the feedback loop of a manipulator to track along edges. In our control system, the data from the tactile sensor is first processed to find edges. The parameters of these edges are then used to generate a control signal to a hybrid controller. In this paper, we present theory for tactile edge detection, and an edge tracking controller. In addition, experimental verification of the edge tracking controller is presented.

1. Introduction

Object recognition is an important problem in robotics [18], particularly for autonomous manipulation systems. In the most general form, it is the problem of determining the environment from sensory data. The long-term goal of our research is to address the issue of object recognition using tactile data through the process of exploring the environment by moving the sensor. We call this approach dynamic object exploration.

Dynamic object exploration involves scheduling moves of the manipulator based on previously acquired data in order to create a more complete description of the object that is being explored. Thus, there is an interaction between manipulation and sensing. In dynamic exploration, the scheduled move affects the data obtained from the sensor, which in turn affects the next move of the manipulator. The two main steps in dynamic object exploration are: first to create strategies for scheduling manipulator moves; and second, to develop processing algorithms that will extract features of interest from the currently available data.

Researchers have actively addressed issues in both of the above mentioned components of dynamic object exploration and especially so in the context of using tactile data for exploration. Early work in edge and surface tracking was done by Bajcsy [2]. In this work, the utility of using a tactile sensor to move about an object to detect features is discussed. Work in object recognition has been done by Allen [1], Dario, et al [7], Ellis [8], Grimson [10], Klatzky, et al [12], Schneiter [19], and Stansfield [20]. Some of these groups [7, 12] take the approach of creating tactile subroutines to find particular features of an object. In this approach, a feature is extracted by calling a specific subroutine that moves and takes the appropriate measurements with the sensor. Other groups have taken a completely different approach to object recognition [8, 10, 19]. They have devised algorithms that determine the best path to approach a planar polygonal object such that it can be identified in a small number of discrete moves of the sensor.

The area of tactile image processing has received less attention than object exploration. Work has proceeded in both pattern recognition [14], and edge finding [16, 9]. Muthukrishnan, et al [16] developed a vision-like algorithm to detect edges in a tactile image. In contrast, Fearing and Binford [9] use the impulse response of their sensor to process the signals to measure the curvature of an object.

At Carnegie Mellon, our research group is addressing multi-sensor based manipulation. The goal of our research is to incorporate position, velocity, force, vision, and tactile sensors in the real-time feedback loop to create an autonomous manipulator system. The focus of this paper is to describe the use of a tactile sensor in the real-time feedback loop for edge tracking. We call this system a dynamic edge extractor. Our methodology utilizes a tactile sensor mounted on the end-effector of a manipulator to obtain data about objects. This system consists of both signal processing and control aspects. The role of the signal processing module is to find edges in the data from the tactile sensor, while the control module generates signals to servo the center of the tactile sensor along the edge. In this paper, we present the theory behind our signal processing and control modules in addition to the results of an experimental verification of the dynamic edge extractor using the CMU Direct Drive Arm II and a Lord LTS-210 Tactile Array Sensor.

2. Signal Processing

In this section, we present a brief description of the signal processing required to detect edges in a tactile image. Further details are presented in [3]. We propose algorithms that are based on the physical properties of the tactile sensor. The important charactersitics of our sensor, a Lord LTS-210, are that it has low spatial resolution and exhibits mechanical cross-talk noise. The noise is due to mechanical coupling generated by the rubber covering on the sensor. In addition, the background tactile elements (taxels) have non-zero force readings due only to mechanical cross-talk. Thus, assuming there is no cross-talk, edges are present at the locations where measured force goes from non-zero to zero. Taking these properties into account, we have devised an edge detecting algorithm that consists of two steps. The first step is an adaptive thresholder to remove cross-talk noise, and the second consists of an edge detector.

2.1. Adaptive Thresholder

The purpose of this filtering stage in our algorithm is to remove the effects of cross-talk noise from the tactile image. This operation simplifies the process of detecting edges because with no cross-talk noise, the locations where the force goes from a non-zero value to zero indicate the edges of planar surfaces. As will be discussed in the following section, the edge detector does not utilize the magnitudes of the taxels. It only uses the state of each taxel, whether it is zero or non-zero. Thus the filter may distort magnitude without adverse side effects. In the ensuing discussion of the thresholding algorithm, we show how this property is utilized.

Tactile images are very noisy. However, the noise of concern exists only at the edges of objects. In particular, the noise causes taxels that should read a force of zero to have a non-zero value. These taxels always have values that are less then their neighbors which are directly beneath the object. Hence, a thresholder that can choose the appropriate threshold at each taxel may be used to remove the noise. The threshold value is determined by the neighbors of the current taxel, thus making the thresholding an adaptive procedure. The proposed algorithm consists of three basic steps:

1. At each pixel, the force value at each of the four-connected neighbors is checked.
2. If any of these neighbors are large enough to have caused the current pixel to be noise (greater than threshold), the current pixel is set to 0 (no force).
3. Otherwise the pixel is set to a constant.

The threshold for a given taxel value is the minimum value that a neighbor must have in order for the original taxel to be cross-talk. Thus, if all neighbors of a taxel are below threshold, the taxel is considered to be part of the signal. Threshold values are determined through an experimental procedure which is described in [3]. Thresholds obtained with our sensor are summarized in Table 2-1. In this table, the first column is the cross-talk value, and the second column is the smallest value that will cause that cross-talk value.

Cross Talk	Minimum Neighbor
2	4
4	10
6	20

Table 2-1: Filter Threshold Values

2.2. Edge Detector

Edge detection in the thresholded tactile image is accomplished very efficiently. This is largely due to the assumption that the measured force goes to zero on one side of an edge, and is some non-zero value on the other side of the edge. Since the thresholding step filters out the taxels that have non-zero readings purely due to cross-talk, all that remains for the edge detector to do is to find those taxels that are neighbors of taxels with zero values.

Our edge detection algorithm consists of the following steps:

1. For each taxel, the eight-connected neighbors are checked.
2. If at least one of these neighbors is 0, the current taxel is copied to the edge image.
3. Otherwise the corresponding taxel in the edge image is set to 0.

This algorithm is very fast and minimally distorts the size, shape and position of the object. What does not come out of the algorithm is an estimate of the slope of the edges. Vision researchers have recognized that slope provides a considerable amount of information about the edge [6, 16]. However, since tactile images are small, they are simple in structure, and simply finding the position of edges appears to be sufficient for higher-level processing. In addition, standard vision edge operators that do provide this information have a number of undesirable characteristics for taction, such as edge spreading and high computational requirements. The slope of object edges may be obtained by combining the tactile and position information as the sensor tracks along the edge of an object.

3. Control

In this section, we discuss the control aspects of dynamic edge extraction [4]. The edge tracker starts on an edge and uses the extracted parameters of the edge to generate control signals to move along that edge. The control scheme is hierarchical, with the tactile controller wrapped around a cartesian space hybrid controller. In the ensuing paragraphs, we describe both the hybrid controller used in our scheme and the tactile controller.

3.1. Hybrid Controller

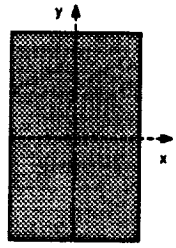

Figure 3-1: Sensor Coordinate Frame

Hybrid force and position control provides the ability to control both forces on the end effector and position of the sensor, [17]. Figure 3-1 depicts the sensor frame coordinate axis. The shaded box shows the face of the sensor. The x and y axis lie in the plane of the sensor, and the z axis (not shown) points out of the page. For tactile sensing, we control the normal force, and torques about the x and y axis of the sensor. Position is controlled in the xy plane, and about the z axis of the sensor. Normal force control is necessary to ensure that the tactile data is within the middle of the operating range [3]. High forces change the sensor cross-talk characteristics, and low forces result in a very low signal to noise ratio. Controlling torques about the x and y axis of the sensor allows tracking of surfaces that are not flat. Specifically, the desired torques are set to zero in order to place the sensor as flush as possible against the surface. Position control in the plane of the sensor is used because the processed sensor data provides information about the surface in the xy plane of the sensor. Thus, it is in this plane that we generate position control signals. Further, we control rotation about the z axis of the sensor. In summary, the hybrid controller commands position/orientation in three degrees of freedom, and commands force/torque in the other three. The x and y positions, and the rotation about the z axis of the end effector are controlled. Torques about the x and y axis, and force along the z axis are controlled.

3.2. Edge Tracking Controller

The edge tracking controller utilizes the edges extracted from tactile images to generate new reference signals for the hybrid arm controller. Edge tracking is initiated by positioning the tactile sensor on an edge. Through the edge detection technique discussed in the preceding section and the Modified Adaptive Hough Transform (MAHT) [5], our implementation of the Hough Transform, the tracker finds the parameters of the edge. The tracker queries a higher level process to determine which direction to travel, and begins to move the end effector in that direction. After this startup, the edge tracker functions independently of higher level input, utilizing a weighted least squares line fit to the data to determine the current parameters of the line. The Hough Transform is also performed every cycle to determine if any

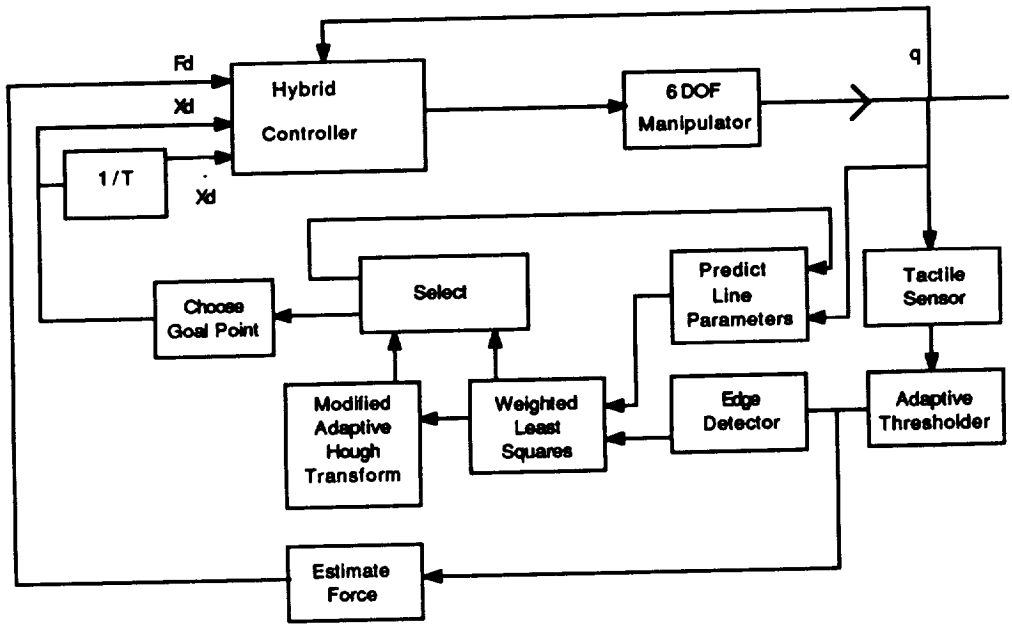

Figure 3-2: Block Diagram of Edge Tracking Controller

new edges have become visible. Each time through the loop, the robot's reference position is set to be the end point of the line segment on the sensor. Thus, if the edge extends past the end of the sensor, the point where the line intersects the edge of the sensor is selected as the goal point. As the end of a edge becomes visible to the sensor, the reference position is set to the actual end of the edge. In addition, a reference velocity is set such that the end effector should arrive at the reference position at the same time that a new reference position is generated.

Now we consider the controller in detail. Figure 3-2 is a block diagram of the edge tracker. Starting at the upper right corner of the diagram, the tactile sensor is mounted at the end effector of the manipulator. The touch image is first thresholded, with the adaptive thresholder algorithm discussed in Section 2. The thresholded image is then sent to both the edge detector and the force estimator.

The *Estimate Force* box computes a reference force such that the taxels operate in the middle of their range. Specifically, it takes the thresholded image and counts the number of taxels that are non-zero. The number of non-zero taxels multiplied by the area of each taxel is an estimate of the area of the sensor that is covered by objects. A desired normal force to the sensor may then be generated by dividing the full scale force by the area in contact with the surface. Full scale force is the total force to drive all taxels to mid-range when the entire sensor is on a flat surface.

Now, we return to the output of the adaptive thresholder. The thresholded image is passed through the edge detector (discussed in Section 2) and the result is sent to a weighted least squares line parameter estimator. This algorithm is used to estimate the slope and intercept of the edge based on the slope and intercept computed in the previous cycle. All data points in the image are weighted with a gaussian function, with $\sigma = 0.75$. A standard deviation of 0.75 was determined from our experimental work to be the best compromise for both accurate line fitting and adapting of line parameters. The weighting function is oriented such that data points located on the predicted location of the line have the highest weight. As the perpendicular distance of a point to the predicted line increases, the weight of that point decreases.

Use of this weighting function allows us to pass all of the data points to the line fitting algorithm without pre-processing to remove points that don't appear to be on the line. After the slope and intercept parameters for the edge are determined, the data points in the image corresponding to that line are removed. Also, the end points of the line are determined at this stage. These computations are the same as those performed by the MAHT, the details of which are discussed in [5]. The point removal and end point computation are part of the *Weighted Least Squares* box in the block diagram.

The weighted least squares computation requires an estimate of the parameters of the previous line segment in the current frame. The *Predict Line Parameters* box in the diagram performs this operation. The end effector will have translated and possibly rotated since the previous set of line parameters were determined. Thus the slope and intercept stored from the previous cycle must be updated to reflect this change. The predictor calculates the parameters of the current line based on the parameters of the previous line, the position of the end effector in the previous cycle, and the current position.

The remaining image is passed on to the Modified Adaptive Hough Transform. The MAHT extracts multiple lines of arbitrary slope from low signal to noise input data. Any line segments other than the one being currently tracked will be detected by this algorithm. If there are no edges remaining in the image, the transform exits, and the parameters and end points determined by weighted least squares are passed through the *Selector*. If there are new line segments, the higher level process will be informed. At this point a new line segment may be selected for tracking. When a new segment is selected, the *Selector* passes the parameters determined by MAHT to the predictor, and the end points determined by MAHT to the *Choose Goal Point* process.

Finally, *Choose Goal Point* determines which of the two end points of the segment should be set as the new reference position for the robot. The choice is made such that the robot continues to move in the same direction that it has been moving. The reference velocity is set to the distance to the new goal position divided by the edge tracking sampling period.

3.3. Discussion

The design of the edge tracking controller has several desirable properties. Specifically, it handles the of ends of segments smoothly, it can track curves in addition to straight lines, and the design is tolerant of any size sensor and data rate. In the following paragraphs, we discuss each of these points in some detail.

As the tactile sensor approaches the end of a line segment, the controller slows the arm down. When the center of the sensor reaches the end point, the arm stops. This action is a natural consequence of the way that new reference points for the hybrid controller are generated. In each cycle, the visible end of the line segment is chosen as the new reference point. Hence, before the end of the line is under the sensor, the point where the line leaves the sensor is the reference point. However, as the end point becomes visible, the controller chooses that point as the goal. This new goal point is closer to the center of the sensor than the edge of the sensor, and as a result, the velocity of the arm decreases. As the center of the sensor gets closer to the end of the segment, the arm continues to slow down, until it stops when the segment end is below the center of the sensor. This allows the arm to accurately position itself at the end of the segment, and provides an easy way to detect the end of a line segment.

Gradual curves appear as piecewise straight lines to the tactile sensor, allowing it to track them. In

each cycle, new line parameters are fit to the segment of the curve that is under the sensor by the weighted least squares method. The parameters that control the weighting are the line parameters from the previous cycle. The old parameters will not be correct, as both the slope and intercept of the new section of the curve may be different. However, the old values are close enough to the correct ones that the weighting function will still be in approximately the correct location, and weighted least squares will extract the correct new parameters. Thus, the procedure of adapting the line parameters each cycle allows the system to track curves in addition to straight lines.

The sampling rate of the sensor only affects the maximum tracking velocity. As discussed above, the reference point for the hybrid controller is set to the intersection of the line with the edge of the sensor. Further, the reference velocity is set to the length of the new reference trajectory divided by the cycle time of the controller, T. As the sampling rate of the sensor decreases, T increases. Thus, desired velocities are reduced, and the reference points are placed closer together. In this scheme, there is no danger of the manipulator traveling faster than new data arrives.

4. Experimental Apparatus

In this section, we describe the hardware used in our laboratory to implement the tactile edge follower. The hardware consists of the CMU DD Arm II, control computers, a Lord Force/Torque sensor, and a Lord LTS 210 Tactile Array Sensor. The tactile control software is run on a Sun 3 computer.

4.1. Control Computers

The hardware of the DD Arm II control system consists of four integral components: the Sun workstation, the Motorola M68000 microcomputer, the Marinco processors and the TMS-320 microprocessor-based individual joint controllers. All of the computers, with the exception of the Sun are connected through a common Multibus backplane. The Eurocard Sun 3 is connected to the backplane through a serial line and interface card, operating at 4800 Baud. A simple packet based communications scheme between the M68000 Coordinating Processor and the Sun operates over this serial connection.

Previous control work included the development of the customized Newton-Euler equations for the CMU DD Arm II which achieved a computation time of 1 *ms* on the Marinco processor. The details of the customized algorithm, hardware configuration and the numerical values of the dynamics parameters are presented in [11]. For tactile sensing, we run a cartesian position controller on one of the Marinco boards, while gravity compensation torques are computed on the other Marinco. The edge tracking controller runs on the Sun. Each cycle, new reference positions are sent from the Sun to the 68000, and the current position is transmitted from the 68000 to the Sun.

4.2. Lord LTS 210 Tactile Array Sensor

To perform our taction experiments, we added a Lord LTS-210 tactile array sensor to the DD Arm II system. This sensor is mounted at the end-effector of the robot. The sensor is an array of 10×16 elements spaced on $1.8mm$ centers [13]. Each sensing site is a small plunger mounted such that as it is depressed, it blocks the light path between a LED and a photodiode [15]. Sixteen different increments in deflection may be read for each site in the sensor. A sheet of rubber protects the top surface of the sensor, but also mechanically couples the sensing sites. The sensor is interfaced to the Sun 3 through a 9600 Baud serial line.

5. Experimental Results with the CMU Direct Drive Arm II

In the ensuing paragraphs, we present the results of two different edge tracking experiments along with some observations about the use of a tactile sensor for edge tracking. First, we discuss a change in the thresholds used by the adaptive thresholder, and our strategy for orienting the tactile sensor for edge tracking. Then, we show the trajectory followed by the manipulator while tracking both straight and curved edges. The straight edge experiment allows us to view the accuracy of the tracking system, while the curved edge experiment shows the line parameter adaptation capability.

5.1. Observations

Our experiments to determine the threshold values for the adaptive thresholder show that taxel values of 2 are noise if there is a four connected neighbor of value 4 or greater [3]. During early edge tracking experiments, however, we found that after the sensor is moved over a surface for a distance of a few centimeters random 2's appear in the image. Thus, motion of the sensor against a surface makes force values of 2 unreliable. To compensate for this phenomena, the adaptive thresholder parameters were adjusted to always filter out twos regardless of the force on neighbors. No side effects in system capability are produced by the elimination of 2 as a usable force value. As discussed in Section 3, forces on the sensor are maintained above 2 for best utilization of the sensor.

We track edges with the sensor oriented such that it only contacts the edge, and not the surfaces of the object. Although the algorithms presented in the previous sections are general and may be used to track edges with the sensor in contact with the surface, we found that the friction between the object and the sensor is very high when the system is used in this mode. With our approach, two effects combine to reduce the friction. First, less area is in contact with the surface since the sensor is only contacting a line, instead of a plane. Second, a lower normal force is required. The normal force necessary to operate the sensor in the mid-region is proportional to the area of the sensor in contact with the surface. Each taxel in contact with the surface must experience a force large enough to keep it in operating range. Thus the normal force that must be exerted by the manipulator is approximately the product of the force each taxel requires and the number of active taxels. Lower forces on the sensor not only help to reduce the requirements placed on the manipulator, but also reduce wear on the sensor.

5.2. Edge Tracking

Figure 5-1 shows the result of tracking a straight edge on a metal box. In each cycle, the position of the end effector was recorded. Dots in the graph correspond to these end effector positions. Thus, the graph shows the distance between samples in addition to the robot's trajectory. The dashed line in the figure is an approximation of the location of the actual edge and is included for reference. This reference line is nearly indistinguishable from the robot's trajectory. In this experiment, the tactile sensor was oriented such that the long dimension (the 16 rows) was parallel to the direction of travel. The end effector traced a path starting at (0.47, 0.1) and ending at (0.72, 0.26), with an average speed of 5 *mm/sec*.

The plot (Figure 5-1) shows the typical characteristics of our edge tracking system. First, we note that its accuracy is acceptable and the errors are within the width of the lines in this plot. The position errors are approximately $1mm$. Remember that the tactile sensor resolution is $1.8mm$, and the reference line is only an approximation to the actual edge. Thus, we conclude that the position error is well within expectations for the system.

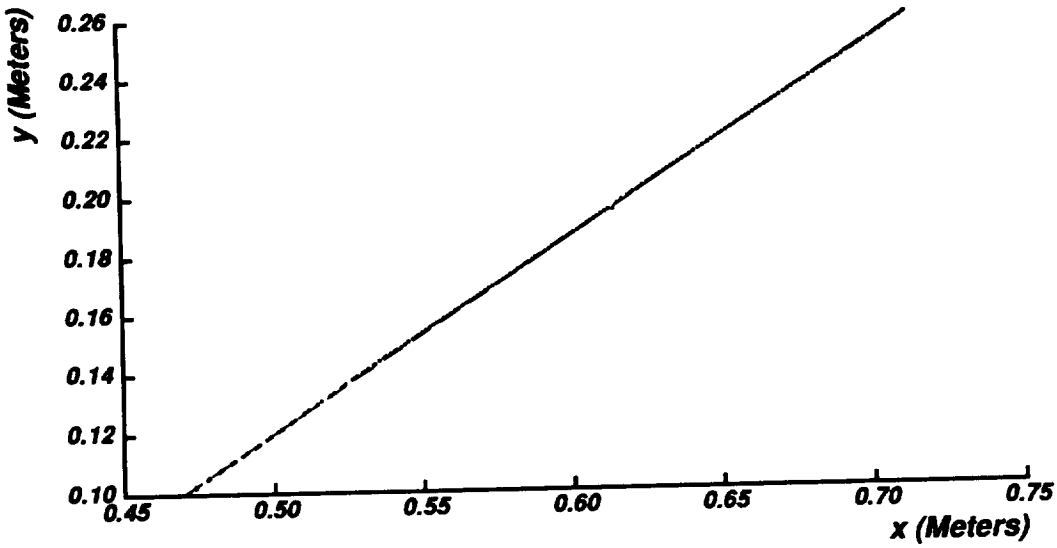

Figure 5-1: Straight Edge Tracking

Now we discuss the start and end points. At the start, (0.47, 0.1), the velocity does not appear to be as consistent as the during the remainder of the trajectory. This is to be expected as the end effector moves to place the center of the tactile array on the line, and the estimated line parameters adapt to the edge. Further, at the beginning of the line the manipulator is at rest. Thus, the first move request is a step input to the cartesian controller. Our current controller is somewhat under-damped and requires time to reach steady motion. On this particular run, the motion of the sensor smoothed out after 4 or 5 *cm.* At the very end of the trajectory, the dots become close together, indicating that the end effector slowed down. This is precisely the action designed into the system. The visible end of the line segment is always chosen as the new goal point. Thus, as the end of an edge comes into view, the commanded trajectory length, and end effector velocity decreases.

The next experiment involved tracking a *S* shaped object. Figure 5-2 shows the results when the sensor is started with the long dimension approximately oriented at a positive 45 degree angle to the x axis. Tracking follows a smooth arc beginning at (0.45, -0.14) and ending at (0.93, 0.21). The primary result from this experiment is the verification of the line parameter adaptation. The edge tracker always attempts to follow a straight line. Curves are taken to be piecewise linear, with line parameters changing slightly each cycle. The motion shown in Figure 5-2 clearly shows that line parameters are adapting properly. As with the straight line, we note a small amount of oscillation at the beginning of the trajectory, and a decrease in velocity at the end.

6. Summary

This paper presents the utilization of a tactile sensor in the feedback loop of a robot controller. There are two main components to our dynamic edge tracker: tactile signal processing and control. We base our tactile signal processing algorithms on the physical properties of the sensor. Thus, we accomplish edge detection by a two step process that first filters mechanical cross-talk noise and second finds edges by looking for transitions from non-zero to zero force. The controller uses detected line segments to generate reference signals for a manipulator. During each cycle of the edge tracker, the estimated

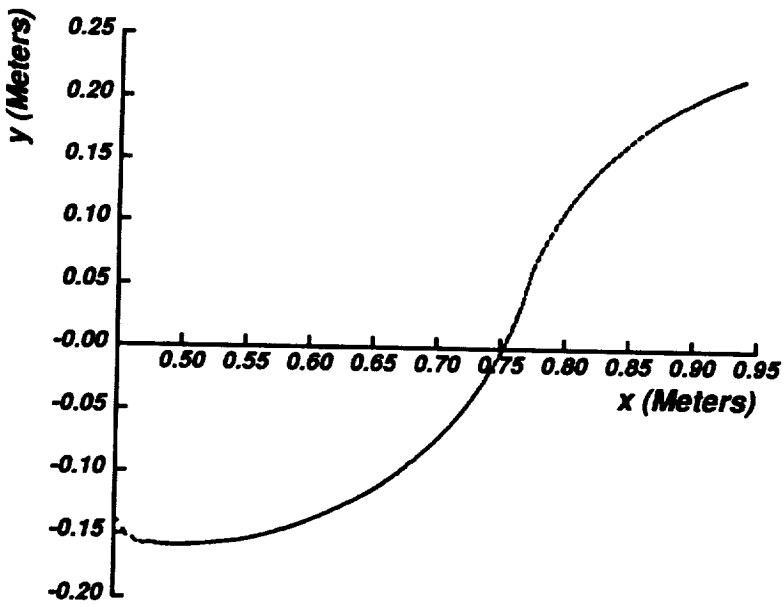

Figure 5-2: *S* Curve Tracking

parameters of the line are transformed to the current frame. These parameters are used to position a weighting function for a weighted least squares estimate of the new line. Performing this procedure every time through the control loop allows the line parameters to continuously adapt. Continuous adaptation of the parameters, in turn, allows the system to track curved objects in addition to straight objects.

References

[1] P. Allen.
Surface Descriptions from Vision and Touch.
Proc IEEE Conf on Robotics and Automation :394 - 397, 1984.

[2] R. Bajcsy.
What can we learn from one finger experiments?
In M. Brady and R. Paul (editor), *The First International Symposium on Robotics Research*. MIT Press, Cambridge, MA, 1984.

[3] A.D. Berger, and P.K. Khosla.
Edge Detection for Tactile Sensing.
Proc. SPIE's Cambridge Symposium on Optical and Optoelectronic Engineering: Advances in Intelligent Robotics Systems , November, 1988.

[4] A.D. Berger and P.K. Khosla.
Real-Time Feature Tracking Using a Tactile Sensor.
Technical Report, Departement of Electrical and Computer Engineering, Dec, 1988.

[5] A.D. Berger, and P.K. Khosla.
The Modified Adaptive Hough Transform (MAHT).
Technical Report, Dept. of Electrical and Computer Engineering, Carnegie Mellon University, 1988.

[6] J.B. Burns, A.R. Hanson, and E.M. Riseman.
Extracting Straight Lines.
IEEE Trans. on Pattern Analysis and Machine Intelligence 8(4):425 - 455, July, 1986.

[7] P. Dario, M. Bergamasco, D. Femi, A. Fiorillo, A. Vaccarelli.
Tactile Perception in Unstructured Environments: A Case Study for Rehabilitative Robotics Applications.
Proc IEEE Conf on Robotics and Automation 3:2047 - 2054, 1987.

[8] R.E. Ellis.
Acquiring Tactile Data for the Recognition of Planar Objects.
Proc IEEE Conf on robotics and Automation 3:1799 - 1805, 1987.

[9] R.S. Fearing, and T.O. Binford.
Using a Cylindrical Tactile Sensor for Determining Curvature.
Proc IEEE Conf on Robotics and Automation :765 - 771, 1988.

[10] W.E.L. Grimson, and T. Lozano-Perez.
Model-Based Recognition and Localization from Tactile Data.
Proc IEEE Conf on Robotics and Automation :248 - 255, 1984.

[11] P.K. Khosla and T. Kanade.
Experimental Evaluation of the Feedforward Compensation and Computed-Torque Control Schemes.
In Stear, E. B. (editor), *Proceedings of the 1986 ACC.* AAAC, Seattle, WA, June 18-20, 1986.

[12] R.L. Klatzky, R. Bajcsy, and S.J. Lederman.
Object Exploration in One and Two Fingered Robots.
Proc IEEE Conf on Robotics and Automation 3:1806 - 1809, 1987.

[13] Lord LTS-210 Tactile Sensor.
Installation and Operations Manual.
Lord Corp., Cary, North Carolina, 1987.

[14] R.C. Lou, and Tsai.
Object Recognition using Tactile Image Array Sensors.
Proc IEEE Conf on Robotics and Automation :1248 - 1253, 1986.

[15] G.A. McAlpine.
Tactile Sensing.
Sensors :7 - 16, April, 1986.

[16] C. Muthukrishnan, D. Smith, D. Myers, J. Rebman, and A. Koivo.
Edge Detection In Tactile Images.
Proc. IEEE Conf. on Robotics and Automation 3:1500 - 1505, April, 1987.

[17] M.H. Raibert and J.J. Craig.
Hybrid Position/Force Control of Manipulators.
J. Dynamic Systems, Measurement, Control , 1981.

[18] A.A.G. Requicha.
Representations for Rigid Solids: Theory, Methods, and Systems.
ACM Computing Surveys 12(4):437 - 464, December, 1980.

[19] J.L. Schneiter.
An Objective Tactile Sensing Strategy for Object Recognition and Localization.
Proc. IEEE Int'l Conf. on Robotics and Automation :1262-1267, April, 1986.

[20] S.A. Stansfield.
Primitives, Features, and Exploratory Procedures: Building a Robot Tactile Perception System.
Proc IEEE Conf on Robotics and Automation 2:1274 - 1279, 1986.